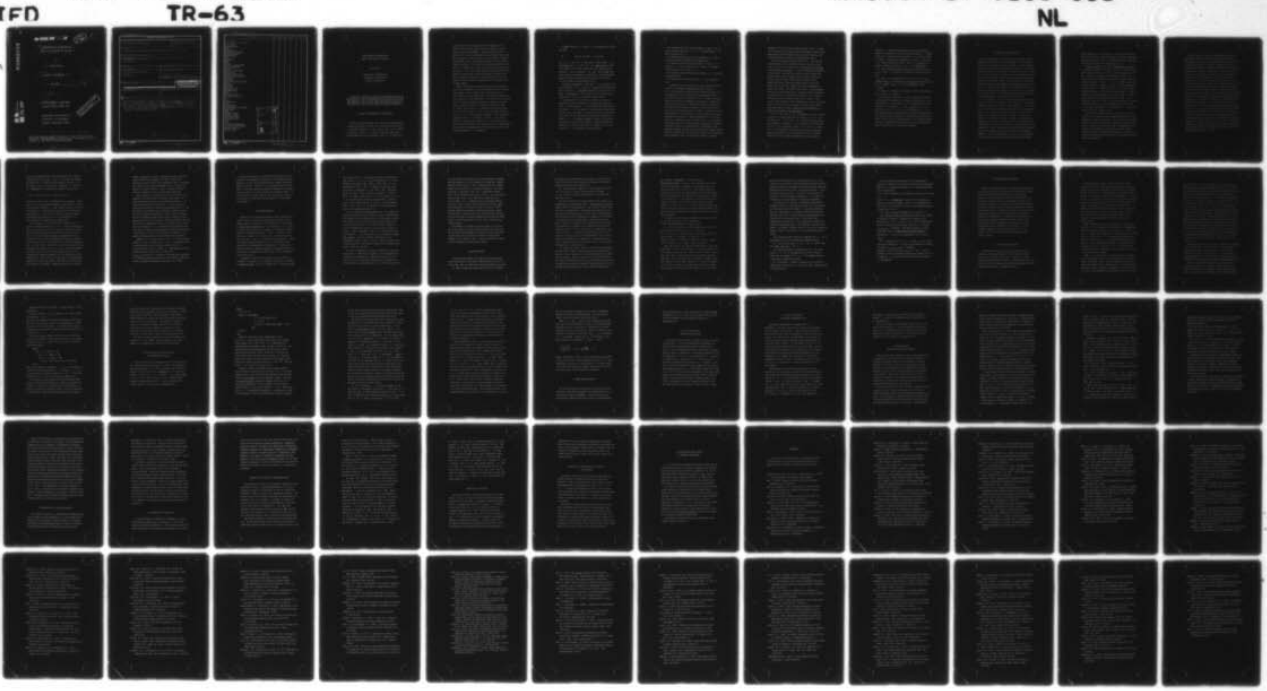
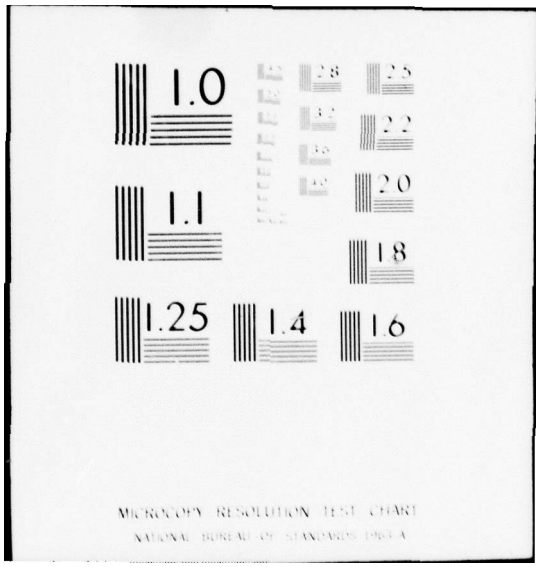
AD-A055 722 WASHINGTON UNIV SEATTLE DEPT OF MATHEMATICS F/G 12/1
COMBINATORIAL OPTIMIZATION: WHAT IS THE STATE OF THE ART.(U)
MAY 78 V KLEE N00014-67-0103-003
UNCLASSIFIED TR-63 NL

1 OF 1
AD
A055 722



END
DATE
FILMED
8-78
DDC


1.0
1.1
1.25
1.4
1.6
2.8
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

12

COMBINATORIAL OPTIMIZATION: WHAT IS THE STATE OF THE ART?

by

Victor Klee

Technical Report No. 63

May 1978 69 p.

TR-63

Contract N00014-67-0103-003

Project Number NR044 353

Department of Mathematics
University of Washington
Seattle, Washington 98195

This reserach was supported in part by the Office of Naval Research. Reproduction in whole or part is permitted for any purpose of the United States Government.

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) University of Washington	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Combinatorial Optimization: What is the State of the Art?

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical

5. AUTHOR(S) (First name, middle initial, last name)

Victor Klee

6. REPORT DATE May 1978	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
8a. CONTRACT OR GRANT NO. N00014-67-A-0103-0003 b. PROJECT NO NR044 353 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report No. 63 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

~~Releasable without limits on dissemination~~

DISTRIBUTION STATEMENT A
Approved for public release; Distribution Unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

This survey attempts, without including many details of algorithms or of the underlying theory, to answer the ~~following~~ questions: What is combinatorial optimization? What are the landmarks of the theory? What are the most exciting recent advances? What are the most promising directions for research? What are the best sources of further information?

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
algorithm						
approximate algorithm						
assignment						
average-case						
base						
bipartite						
branch-and-bound						
combinatorics						
complexity						
cut						
diameter						
DTM						
flow						
Floyd's algorithm						
good algorithm						
graph						
greedy algorithm						
heristic						
independent set						
linear programming						
matroid						
matching						
matroid intersection						
maxim						
minimzat						
network						
network flow						
NP NPC NPI						
NTM						
optimization						
oriented matroid						
P						
path						
polyhedral						
path						
polyhedron						
polynomially bounded						
polytope						
RAM						
random access						
random graph						
shortest path						
spanning tree						
tour						
tree						
totally unimodular						
travelling salesman						
turing machine						
worst case						

COMBINATORIAL OPTIMIZATION:
WHAT IS THE STATE OF THE ART?

Victor Klee

Department of Mathematics

University of Washington

Seattle, Washington

This survey attempts, without including many details of algorithms or of the underlying theory, to answer the following questions: What is combinatorial optimization? What are the landmarks of the theory? What are the most exciting recent advances? What are the most promising directions for research? What are the best sources of further information?

1. WHAT IS COMBINATORIAL OPTIMIZATION?

Many practical problems, especially those from operations research and computer science, are concerned with optimizing a real-valued function f over a finite set x of d-tuples of integers. Often x is not presented explicitly but is defined implicitly in some manner. When f is linear and x

is defined by a finite number of linear inequality constraints with integer coefficients, the problem is one of (linear) integer programming. Starting with the work of Gomory (1958), finite algorithms and an extensive theory have been developed for such problems. See Hu (1969), Garfinkel and Nemhauser (1972), Geoffrion and Marsten (1972), Balinski (1974) and Hammer et al. (1977) for details and further references. Computational experience with these algorithms has been mixed. Some fairly large problems have been solved, but experience often shows, even with problems of modest size, that an algorithm requiring a very large finite number of steps may have no practical advantage over one requiring an infinite number.

Most of combinatorial optimization deals with problems that can be formulated as integer programs but have an underlying combinatorial structure that lends itself to the development of special algorithms. One naturally hopes that these algorithms, when applicable, will be more efficient than general integer programming algorithms. Many of the problems are associated with a directed or undirected graph $G = (N,A)$ and a function α defined on the (finite) node-set N or the arc-set A or on $N \cup A$ and taking values in $]-\infty,\infty]$. (G,α) is often called a network. It is assumed for simplicity that G is loopless, so that A is a set of ordered or unordered (according as G is directed or not) pairs of distinct nodes. Arcs may be written as ordered pairs (i,j) with the understanding that they will be interpreted as unordered when G is undirected.

A path from node s to node t is a sequence of arcs of the form

$$(*) \qquad (x_0,x_1),\ (x_1,x_2),\ \ldots,\ (x_{k-1},x_k)$$

with $x_0 = s$ and $x_k = t$. When such a path exists, t is accessible from s. A path $(*)$ is a simple path if there is no repetition of nodes (other than that implied by the notation). It is a circuit if $x_k = x_0$, and a simple circuit if there is otherwise no repetition. Note that simple paths and simple circuits may conveniently be regarded as sets (rather than sequences) of arcs.

A graph is connected if each node t is accessible from from each node $s \neq t$. A tree is a connected graph that contains no circuit. A spanning subgraph is one that uses all nodes. A matching is a set of arcs no two of which have a common node. A tour is a spanning simple circuit.

For purposes of illustration, we focus here on five problems of network optimization, assuming for simplicity that A is the domain of α and (except where the contrary is stated) $\alpha \geq 0$. The graph G should be undirected in (2) and (4), but in the others it may be directed or undirected. In all but (3), the desired solution may be regarded as a subset of A, the length or weight of such a set being the sum of the α-values of its members. (Thus the objective function is a linear function of arc lengths.) In (3) the values of α are regarded as flow capacities.

(1) (Minimum path) For two given nodes s and t of G, find a shortest path from s to t or conclude that t is not accessible from s.

(2) (Minimum spanning tree) For a connected G, find a connected spanning subgraph of minimum weight.

(3) (Maximum flow) For two given nodes s and t of G such that t is accessible from s, find an α-feasible flow of maximum value from s to t.

(4) (Maximum matching) Among all matchings in G, find one of maximum weight.

(5) (Minimum tour) Given a tour in G, find a shortest tour.

In addition to the many practical optimization problems for which the immediate mathematical model is one of (1) - (5), many other problems can be reduced to these or to a combination of them. The reader is undoubtedly faced, each day, with several instances of (1). Problem (2) arises in the construction of communication networks. Problem (3) stems from an attempt to evaluate the capacity of the Eastern European rail network to support a large-scale conventional war (see Billera and Lucas, 1976); it and its relative, the minimum-cost flow problem, are used to model a variety of transportation problems. Though algorithms for (4) are perhaps not as obviously useful as those for (1) - (3), matching algorithms have in fact been applied to problems of personnel assignment, pairing of machined parts, scheduling of two-processor systems, various routing problems (see Lawler (1976) and his references for all of these), evaluation of biomedical data (Tanimoto, 1976), and finding the rank of a matrix

(Anderson, 1975; Klee and van den Driessche, 1977). Problem (5) is a form of the famous traveling salesman problem, whose applications include computer wiring, vehicle routing, and job shop scheduling (Lenstra and Rinnooy Kan, 1975).

For the many direct applications of (1) - (4), and for the reduction of other problems to these, the best single source is the book of Lawler (1976), Combinatorial Optimization: Networks and Matroids. It will be mentioned frequently in the sequel. The books or survey articles of Berge and Ghouila-Houri (1965), Bradley (1975), Busacker and Saaty (1964), Dantzig (1963), Eiselt and von Frajer (1977), Elmaghraby (1970), Ford and Fulkerson (1962), Frank and Frisch (1972), Fulkerson (1966), Garfinkel and Nemhauser (1972), Hu (1969), Iri (1969), Karp (1975b), and Whitehouse (1973) also contain much useful information about combinatorial optimization. For problems reducible to (5) the basic references are Cook (1970) and Karp (1972, 1975a). Aho et al. (1974) contains a good introduction to this fascinating subject, and by far the most complete study is the book of Garey and Johnson (1978?), Computers and Intractability: A Guide to the Theory of NP-completeness.

Let $n = |N|$ and $a = |A|$, the numbers of nodes and arcs respectively. When convenient, we assume $N = \{1,2,\ldots,n\}$. In each of (1) - (2) and (4) - (5), the desired solution may be regarded as a subset of A or, relative to a given indexing of A, as an ordered a-tuple of 0's and 1's. There are 2^a such subsets, but many may be quickly excluded by paying some attention to the combinatorics of the problem. However, even after such exclusion and even for moderate

values of n, explicit investigation of all the remaining subsets is impossible even on the fastest computer. To illustrate this, consider the case in which $\alpha > 0$ and the graph $G = (N,A)$ is undirected and complete, so that $A = n(n-1)/2$.

(1) Shortest paths are simple. When G is complete the number of simple paths from s to t is $\Sigma_{k=0}^{n-2} \binom{n-2}{k} k!$.

(2) Each connected spanning subgraph of minimum weight is a tree. When G is complete the number of spanning trees is n^{n-2}.

(4) When G is complete there are $\lfloor n/2 \rfloor$ arcs in each maximum matching. The number of matchings consisting of $\lfloor n/2 \rfloor$ arcs is $\frac{1}{2}\binom{2k}{k}k^k$ when $n = 2k$ and $\frac{2k+1}{2}\binom{2k}{k}k^k$ when $n = 2k+1$.

(5) When G is complete the number of tours (considered as sets of edges) is $(n-1)!/2$.

If a computer required only 10^{-10} seconds (one-tenth of a nanosecond) to investigate a particular spanning tree, the time required to investigate all n^{n-2} spanning trees would be about .01 seconds for $n = 10$, 54 hours for $n = 15$, 8.3×10^5 years for $n = 20$, and 4.5×10^{14} years for $n = 25$. This explains why one speaks of the "combinatorial explosion" in connection with such problems, and why an algorithm that is merely "finite" may be useless except for very small problems. The goal of combinatorial optimization is to find algorithms that are useful even for very large problems.

2. WHAT IS A GOOD ALGORITHM?

How is the speed of an algorithm to be measured? One may compare the performance of two algorithms for the same problem on a few instances of special interest, but how relevant is that to their performance on other instances? One may base the comparison on a large number of instances chosen in a regular or random fashion, but that may be very expensive and yet not provide a reliable indication of performance on instances much larger than those tested. Here we have used (and henceforth will use) the terms problem and instance in the sense of Aho et al. (1974). Each instance is associated with particular numerical data and a problem is the class of all instances of a specified form. For example, the maximum matching problem is the class of all instances of the form (4).

It is expected, of course, that large instances will usually be solved more slowly than small instances. For problems concerning a graph $G = (N,A)$, the parameters $n = |N|$ and $a = |A|$ provide a natural measure of the size of an instance. When τ is a function of these parameters, an algorithm is said to be of (time) complexity $O(\tau(n,a))$ if there is a constant c such that for all $G = (N,A)$ the algorithm requires at most $c\tau(n,a)$ computational steps. The notion of "step" must be interpreted in terms of an appropriate model of computation, and for relevance to practice the model should be based on the random access operations of modern electronic computers. The discussion here is in terms of the RAM model of random access computation described by Cook and Reckhow

(1973) and on pp. 5-14 of Aho et al. (1974), using the uniform cost criterion. Without too much distortion, the reader may simply interpret "step" as a single arithmetic operation (addition, multiplication, comparison, etc).

A good algorithm is one that is polynomially bounded--is of complexity $O(n^p a^q)$ for some p and q. And of course we'd like the exponents p and q to be as small as possible. This notion, popularized by Edmonds (1965a) and Cobham (1965), is meaningful in theoretical studies and also useful in practice. Since $a \leq n^2$, an $O(n^p a^q)$ algorithm is also $O(n^{p+2q})$. However, n and a are mentioned separately because a is much less than n^2 for most graphs arising in practice. Note that if an algorithm is to take advantage of the sparseness of the input graph $G = (N,A)$ in order to achieve performance better than $O(n^2)$, then G cannot be input as its full $n \times n$ adjacency matrix. Instead, the input may consist of lists that tell, for each node $i \in N$, which other nodes j are adjacent to i, and that also give the arc lengths $\alpha(i,j)$.

The above notion of complexity involves the worst-case behavior of an algorithm, but average-case behavior is also important in many applications. For this notion to be meaningful, the sample space must be carefully defined, and for it to be useful the sample space must be appropriately related to the instances in which the algorithm is to be applied. Section 6 discusses the average-case behavior of certain algorithms, but otherwise we are concerned only with worst-case behavior.

In the interest of brevity, the preceding three paragraphs have glossed over some important practical and philosophical questions concerning the encoding of data, the relationship between optimization problems and decision problems, the choice of a model for computation, the reasons for emphasizing polynomial boundedness, and the pitfalls associated with average-case behavior. These matters are treated by Cook and Reckhow (1973), Aho et al. (1974), Lawler (1976), Weide (1977) and Garey and Johnson (1978?). It should also be stressed that if an algorithm works well in practice, frequently handling large instances with amazing ease, then it is of course a good and useful algorithm even though it may not be "good" in the technical sense used here. That is true, for example, of the simplex algorithm of linear programming. Its worst-case behavior is not polynomially bounded (Klee and Minty, 1972; Jeroslow, 1973; Avis and Chvátal, 1976), even for minimum cost flow problems (Zadeh, 1973), and its average-case behavior has not been proved to be polynomially bounded in any sense that is convincingly related to computational practice. Nevertheless, there is strong empirical evidence that the average-case behavior is polynomially bounded (Dantzig, 1963, p. 160). On the other hand, special network algorithms are apt to be faster than the simplex algorithm for problems to which they too can be applied (Bradley, 1975, p. 229).

3. WHAT IS THE STATE OF THE ART?

Problems (1) - (5) are admittedly special in nature, but combinatorial optimization is concerned with special problems. Thus it may be reasonable to judge the state of the art by what is known about (1) - (5). These problems appear in various ways in the next four sections, in which the state of the art is represented by three each of primary landmarks, secondary landmarks, exciting recent advances, and directions for future research. If permitted to make a summary judgment in architectural terms, we might say that the subject of combinatorial optimization has gradually been transformed from the rococo to the merely baroque (thus reversing the evolution that occurred in architecture), but is still far from the simple elegance of the classic orders or the sleek functionalism of modern architecture. This refers to a definition of rococo as "a meaningless assemblage of scrolls and crimped conventional shellwork, wrought into all sorts of irregular and indescribable forms," and of baroque as "odd, grotesque, bizarre, having unusual formation." The scrolls and crimped conventional shellwork are the many papers in the subject which repeat, in only slightly different form, algorithms that appear in earlier papers. As the subject has developed, these "scrolls" have fallen away (been forgotten) and interrelationships have been discovered among much of what remains. The subject has even achieved a considerable degree of architectural unity and may now properly be regarded as a branch of mathematics, or of

operations research, or of computer science. However, the branch is certainly of "unusual formation," because in the long search for "good" algorithms for (1) - (5), none has been proved "best" relative to the RAM model of computation.

4. THREE PRIMARY LANDMARKS

Three lines of investigation stand out for their intrinsic depth and their influence on the rest of the field. They are associated with problem (3), with problem (4), and less directly with (5). We consider only the directed case of (3) because its formulation is slightly simpler.

A. The Maximum Flow Problem and Its Ramifications

For each real function ϕ on A and each node $k \in N$, let

$$\mu_k(\phi) = \Sigma_{(k,j)\in A}\, \phi(k,j) - \Sigma_{(i,k)\in A}\, \phi(i,k).$$

The function ϕ is an (s,t)-flow if it satisfies the conservation conditions

(a) $\mu_k(\phi) = 0$ for all $k \in N \sim \{s,t\}$

(at any intermediate node, the outflow equals the inflow), from which it follows that $\mu_s(\phi) = -\mu_t(\phi)$ (the net outflow at the source s equals the net inflow at the sink t). The number $\mu_s(\phi)$ is called the s-value of ϕ. The flow ϕ is α-feasible if it satisfies the capacity constraints

(b) $0 \le \phi(i,j) \le \alpha(i,j)$ for all $(i,j) \in A$.

Problem (3) asks for an α-feasible flow of maximum s-value. This is a linear programming problem in the variables $\phi(i,j)$ with objective function μ_s. The feasible region is the convex polytope P_α defined by the equalities (a) and the inequalities (b). Thus (3) can be solved by general linear programming algorithms, but it is also amenable to the special methods of combinatorial optimization.

An (s,t)-cut is a set of arcs whose removal from G leaves no directed path from s to t. The initial question about rail capacities led to the max-flow min-cut theorem of Ford and Fulkerson (1956) (discovered also by Elias et al., 1956) and to their book on Flows in Networks (Ford and Fulkerson, 1962), which stimulated much research on combinatorial optimization. The theorem asserts that the maximum of the s-values of the α-feasible (s,t)-flows is equal to the minimum of the weights of the (s,t)-cuts. The flow-augmenting paths used to prove this theorem are used also in an algorithm for finding a maximum flow. As described by Ford and Fulkerson (1957, 1962), the algorithm is not polynomially bounded but it has been applied successfully to a large number of practical problems. Edmonds and Karp (1972) use the

method to produce an $O(na^2)$ algorithm (see pp. 116-120 of Lawler (1976) for an exposition of their work), and Kinariwala and Rao (1977) have a different $O(n^5)$ algorithm. The $O(n^2a)$ maximum flow algorithm of Dinic (1970) is improved by Karzanov (1974) to $O(n^3)$. Section 6 contains additional comments on the Dinic-Karzanov algorithm.

Space permits us to mention only a few of the many consequences and ramifications of network flow theory. An early observation was that if the capacity function α is integer-valued then all vertices of P_α have integer coordinates and hence any linear integer program over P_α can be solved as an ordinary linear program, without worrying about the integrality constraints. That led to the search for other integer programs which could be solved as linear programs and thus to the study of totally unimodular matrices. See Garfinkel and Nemhauser (1972), Hu (1969), Lawler (1976) and their references. In a different but related direction, the theory of blocking and antiblocking polyhedra (Fulkerson, 1971; Chvátal, 1976, pp. 316-318) may be regarded as a far-reaching generalization of the max-flow min-cut theorem.

Suppose, in (3), that each arc $(i,j) \in A$ has not only a capacity $\alpha(i,j)$ but also a cost $\gamma(i,j)$ of sending a unit of flow along the arc, and suppose there exists an α-feasible flow of given s-value v. The minimum-cost flow problem asks for such a flow ϕ whose cost $\Sigma_{(i,j)\in A} \gamma(i,j)\phi(i,j)$ is minimum. The primal-dual approach to finding minimum cost flows led to the general primal-dual linear programming algorithm of Dantzig et al. (1956).

Many other problems of combinatorial optimization can be reduced to the maximum flow problem or the minimum-cost flow problem. See the next paragraph for one example and Lawler (1976) for others. Plainly if a single publication is to be named as the most important landmark of combinatorial optimization, it must be the book of Ford and Fulkerson (1962). It is now outdated in many respects, but that merely indicates its success in stimulating the development of a new branch of mathematics.

B. Maximum Matching

Because of its close relationship to other combinatorial topics, special attention has been paid to the case of the maximum matching problem (4) in which each arc is of weight 1 (cardinality weighting) and G is bipartite. Let P and Q denote the two parts of the node set, and form a directed graph D by adding new nodes s and t, new arcs (s,p) and (q,t) for all $p \in P$ and $q \in Q$, and directing all old arcs from P to Q. Let each arc have capacity 1. Then the maximum matching problem for G is easily seen to be equivalent to the maximum flow problem for D. Hence the maximum cardinality matching problem for bipartite graphs can be solved as a maximum flow problem, but it can be solved in other ways as well.

A theorem of Berge (1957) asserts that for cardinality weighting, a matching M is maximum if and only if it admits no augmenting path. This is a simple path S whose end nodes

are not covered by M and whose arcs belong alternately to $A \sim M$ and to M. If $S \subset A$ is such a path and M' is the symmetric difference $M \oplus S = (M \sim S) \cup (S \sim M)$ then M' is a matching with $|M'| = |M| + 1$. Suppose, conversely, that there exists a matching M' with $|M'| > |M|$. Following Norman and Rabin (1959), consider the graph $G' = (N, A')$ where $A' = M \oplus M'$. Each node of G' is of valence ≤ 2, so G' is a node-disjoint union of isolated nodes, circuits whose arcs alternate between M and M', and alternating paths. Since $|M'| > M$, at least one of the alternating paths is an augmenting path for M.

For the cardinality weighted bipartite case, augmenting paths are relatively easy to find; an $O(na)$ maximum matching algorithm that is unusually easy to understand, analyze, and program can be obtained by specializing the method of Desler and Hakimi (1969). By finding several augmenting paths simultaneously, Hopcroft and Karp (1973) produce an elegant $O(n^{1/2}a)$ algorithm. When G is bipartite but the arc-weighting α is unrestricted, (4) is called the assignment problem. The best available algorithms are of complexity $O(pq^2)$, where p and q are the cardinalities of the two parts of the node-set (Kuhn, 1955; Tomizawa, 1972; Lawler, 1976, pp. 201-207).

The work of Edmonds and others on nonbipartite matching constitutes a second primary landmark in the development of combinatorial optimization. Augmenting paths play a role here too, but it is much more difficult to find them efficiently than in the bipartite case. For several years there

was doubt about the existence of a good nonbipartite maximum matching algorithm, even for cardinality weighting. However, Edmonds (1965a) found an $O(n^4)$ algorithm and then (1965b) showed how to extend it to arbitrary weightings. His implementation is improved in the $O(n^3)$ algorithms of Gabow (1973, 1976) and Lawler (1976, pp. 217-263). See also Feicht, Heck and Pape (1977). For cardinality weighting only, a fairly simple $O(na)$ implementation is provided by Kameda and Munro (1974), and the ideas of Hopcroft and Karp (1973) are extended by Even and Kariv (1975) to yield an $O(n^{5/2})$ maximum cardinality matching algorithm for general graphs. It seems safe to say the latter has not been widely checked, for the joint paper is quite condensed and the detailed exposition, in the dissertation of Kariv (1976), requires more than 200 pages!

The nonbipartite maximum matching problem is certainly one of the deepest for which good algorithms are known, and the rationale behind Edmonds's approach has been influential in other ways that space does not permit us to describe (Edmonds, 1970; Chvátal, 1973). A book on matching is being written by Edmonds and Pulleyblank (1978?).

C. NP-Completeness

A third primary landmark of combinatorial optimization was the discovery by Cook (1970) and Karp (1972, 1975a) of a large class NPC of combinatorial decision problems such that

(i) NPC includes many important and difficult problems

of combinatorial optimization (for example, a close relative of (5))--problems for which good algorithms have vainly been sought for years by many researchers;

(ii) many problems in NPC are at least superficially similar to problems for which good algorithms are known;

(iii) if even one member of NPC admits a good algorithm then they all do.

The precise definition of NPC depends on the notions of a deterministic Turing machine (DTM), a nondeterministic Turing machine (NTM), the complexity of a TM, and other notions from the theory of computation. The details are complicated, but some may be omitted because it is known that a problem admits a good algorithm for the RAM model of computation if and only if it admits one for the DTM model. Thus for present purposes a DTM may be regarded as an ordinary computer program, and an NTM may be regarded as an animate DTM so fecund and imaginative that it can, at the start of a computation, make a finite number k of guesses and immediately split into k replicas of itself (to be run simultaneously), one to investigate the consequences of each guess. The complexity of a TM is a function that tells how long it takes to process sets of input data of various sizes.

Let P ⟨resp. NP⟩ denote the class of all decision problems solvable by DTM's ⟨resp. NTM's⟩ of polynomially bounded complexity. Then P consists of all decision problems that admit good algorithms. And NP consists of all decision problems for which an affirmative answer can be obtained (when correct) by applying a polynomially bounded "checking algorithm" to the appropriate guess output by a finite "guessing algorithm."

(For example, $(x \vee y \vee \bar{z}) \wedge \bar{x} \wedge \bar{y}$ is satisfiable but $\bar{x} \wedge (\bar{y} \vee z) \wedge (x \vee \bar{z}) \wedge (x \vee y \vee z)$ is not.) It is unknown whether there is a good algorithm for deciding the satisfiability of an arbitrary expression of length n. Cook (1970) showed the satisfiability problem to be $\underset{\sim}{NP}$-complete and then Karp (1972, 1975a) established the $\underset{\sim}{NP}$-completeness of many other problems Y by showing that $Y \in \underset{\sim}{NP}$ (usually easy) and that the satisfiability problem, or some other problem known to be $\underset{\sim}{NP}$-complete, is reducible to Y. Garey and Johnson (1978?) have not only the most extensive list of $\underset{\sim}{NP}$-complete problems, but also an excellent discussion of various strategies for proving $\underset{\sim}{NP}$-completeness.

In addition to the decision problem most closely related to (5), the following is also $\underset{\sim}{NP}$-complete.

(5') Determine whether G admits a tour.

It is easy to reduce (5') to the special case of (5) in which all arc-lengths are 0 or 1. For let $A' = \{(1,2),(2,3),\ldots,(n-1,n),(n,1)\}$ and consider the augmented graph $G' = (N, A \cup A')$. Let each arc in A ⟨resp. $A' \sim A$⟩ be of length 0 ⟨resp. 1⟩. Then G admits a tour if and only if G' admits a tour of length 0.

Note that if even one $\underset{\sim}{NP}$-hard problem admits a good algorithm, then $\underset{\sim}{P} = \underset{\sim}{NP}$. In particular, $\underset{\sim}{P} = \underset{\sim}{NP}$ if and only if (5') admits a good algorithm. These facts, (i) above, and the fact that NTM's seem intuitively to be much more powerful than DTM's, are regarded by many researchers as almost conclusive evidence that $\underset{\sim}{P} \neq \underset{\sim}{NP}$. However, it is also conceivable that (5) or (5') admits a good algorithm but does not admit one of complexity less than $O(n^p)$ where p is very large

and the algorithm therefore so complicated that it is unlikely ever to be found. Good algorithms have been useful in practice because for problems of practical interest they have seldom been worse than $O(n^5)$ and the multipliers c have not been excessively large. Does this say more about the complexity of the interesting problems in $\underset{\sim}{P}$ or about the limitations of human ingenuity in devising algorithms?

We mention only a few more $\underset{\sim}{NP}$-complete combinatorial problems, mainly to illustrate the aspect (ii) stated above and to prepare for later comments. In precise treatments, the term "$\underset{\sim}{NP}$-complete" is usually reserved for decision problems such as (5') and not applied to optimization problems such as those stated below. That distinction is ignored here, for our aim is only to provide a rough understanding of $\underset{\sim}{NP}$-completeness. The problems listed below all appear (in slightly different forms) in the list of Garey and Johnson (1978?), and all but (2') are given by Karp (1972, 1975a).

The following problem should be compared with (1). It is $\underset{\sim}{NP}$-complete even in the cardinality weighted version.

(1') For two given nodes s and t of G, find a longest simple path from s to t.

The following should be compared with (2). For each fixed $d \geq 4$, it is $\underset{\sim}{NP}$-complete even in the version in which all arc weights are 0 or 1. The problem arose in the design of telephone networks.

(2') Given that G admits a spanning tree in which no simple path has more than d arcs, find such a spanning tree of minimum weight.

For the cardinality weighting, (4) is equivalent to asking for a smallest set of arcs that covers all nonisolated nodes of G. Though (4) belongs to $\underset{\sim}{P}$, the following problem is $\underset{\sim}{NP}$-complete.

(4') Find a smallest set of nodes that intersects all arcs of G.

A set $X \subset N$ is <u>independent</u> if no two of its members are adjacent (joined by an edge). Since X is independent if and only if $N \sim X$ intersects all arcs, the following problem is equivalent to (4').

(4'') Find a largest independent set of nodes in G.

The assignment problem, mentioned earlier, can be stated as follows in terms of a nonnegative $n \times n$ matrix (α_{ij}): Among all sets S of n pairs (i,j) that include at most one pair from each row and column, find an S for which $\Sigma_{(i,j) \in S} \alpha_{ij}$ is maximum. This problem admits an $O(n^3)$ algorithm, but the 3-<u>dimensional assignment problem</u> is $\underset{\sim}{NP}$-complete. It asks, for a nonnegative $n \times n \times n$ array (α_{ijk}):

(4''') Among all sets S of n triples (i,j,k) such that no two members of S agree in any coordinate, find one for which $\Sigma_{(i,j,k) \in S} \alpha_{ijk}$ is maximum.

Suppose, in fact, that each α_{ijk} is 0 or 1 and the problem is merely to determine whether the maximum in (4''') is equal to n. Even that restricted form of the problem is $\underset{\sim}{NP}$-complete.

5. THREE SECONDARY LANDMARKS

Among the NP-hard problems, the traveling salesman problem (5) and its relatives stand out for the amount of attention they have received and the variety of practical situations in which they have arisen. The same is true of the shortest-path problem (1) among those for which good algorithms are known. The collections of algorithms dealing with these problems form two secondary landmarks of combinatorial optimization. A third is the study of optimization in independence systems, which originated from the minimum spanning tree problem (2). A fourth, not discussed here, is the part of polyhedral combinatorics that studies interrelationships among combinatorial optimization, linear programming duality, and the facial structure of convex polyhedra (Edmonds, 1965, 1970; Fulkerson, 1971, 1973; Lovász, 1972, 1977; Chvátal, 1973, 1975; Balinski and Hoffman, 1978?).

A. The Traveling Salesman

Some important papers from the early history of the traveling salesman problem are those of Eastman (1958), Dantzig et al. (1959), Little et al. (1963), Lin (1965), Gomory (1966) and Shapiro (1966). A comprehensive survey of work up to 1968 is provided by Bellmore and Nemhauser (1968). The present situation is summarized below.

Though Gilmore and Gomory (1964), Sysło (1973) and Garfinkel (1977) have found good algorithms for some useful special cases of (5), developments of this sort are limited by the fact that (5) is NP-hard even for the special case in which G is undirected and (a) each arc-length is 0 or 1, or (b) G is complete and its node-set lies in the Euclidean plane, the length of each arc being the distance between its nodes. The second result is due to Garey, Graham and Johnson (1976) and Papadimitriou (1977). The first follows from NP-completeness of the undirected version of (5'), which holds even for the special case in which G is a 3-valent 3-connected planar graph (Garey, Johnson and Tarjan, 1976).

When a problem calls for minimization of a nonnegative function f, an algorithm for the problem is ε-approximate if each "solution" output by the algorithm has f-value at most $1+\varepsilon$ times the optimum value. Now suppose that $P \neq NP$ and A_ε is (for some ε) an ε-approximate algorithm for the undirected case of (5). Then Sahni and Gonzalez (1976) show A_ε is not polynomially bounded, while Papadimitriou and Steiglitz (1977) show that if A_ε is a local search algorithm of the sort described below, even the search phase is not polynomially bounded.

When G is undirected the traveling salesman problem may be approached by the local search methods of Lin (1965), Reiter and Sherman (1965), Karg and Thompson (1964) as improved by Raymond (1969), and many others. See Bellmore and Nemhauser (1968), Garfinkel and Nemhauser (1972), Savage et al. (1976), and Papadimitriou and Steiglitz (1977) for a few

more references. In these algorithms, one starts with an arbitrary tour and then, by searching a set of tours which are considered its neighbors, either finds a shorter tour and uses it as a new starting point or, if no shorter tour is found in this way, terminates. (For example, the neighbors of a tour T might be all tours T' obtainable from T by replacing three successive arcs $\{w,x\}$, $\{x,y\}$ and $\{y,z\}$ of T by the arcs $\{w,y\}$, $\{y,x\}$ and $\{x,z\}$.) The resulting tour is locally optimal relative to the neighborhood system underlying the search procedure. Since the locally optimal solutions often turn out to be globally optimal or close to that, and since the computation time is often modest, these methods are frequently used in attacking large instances of (5). Intrinsic limitations are indicated by the theorem of Papadimitriou and Steiglitz (1977) and by the fact that even if $\underset{\sim}{P} = \underset{\sim}{NP}$ no exact (0-approximate) local search method can be polynomially bounded (Savage et al., 1976). See Golden (1977) for statistical analysis of another heuristic approach to (5).

Christofides (1976) has an $O(n^3)$ $\frac{1}{2}$-approximate algorithm for the special case of (5) in which G is a complete undirected graph whose arc-lengths satisfy the triangle inequality $(\alpha(i,j) \leq \alpha(i,k) + \alpha(k,j))$. It first finds a minimum spanning tree and then solves a matching problem. His analysis is sharpened by Cornuejols and Nemhauser (1978).

When G is undirected and one seeks a tour that is definitely optimal, the approach of Held and Karp (1970, 1971) is called for. It combines a branch-and-bound procedure (for surveys of such procedures, see Lawler and Wood,

1966; Agin, 1966; Mitten, 1970; Geoffrion and Marsten, 1972; and Garfinkel, 1978) with an ascent method that involves 1-trees, where these consist of a tree with node-set $\{2,\ldots,n\}$ together with two arcs incident to node 1. Since tours are simply 1-trees in which each node is of valence 2, a minimum 1-tree that is a tour is in fact a minimum tour. The ascent method is based on the fact that minimum 1-trees are easy to find, and that certain transformations of arc-length preserve minimum tours but may produce new minimum 1-trees. The approach of Held and Karp has been refined by Held et al. (1974), Helbig Hansen and Krarup (1974), and Smith and Thompson (1977).

For the traveling salesman problem (5), unlike the shortest path problem (1), transition from the "symmetric" (undirected) case to the general "asymmetric" (directed) case results in a significant reduction in the size of instances that can be solved in a reasonable amount of time. For the general case, the best exact method is apparently a modification, due to Smith et al. (1977), of earlier methods of Eastman (1958), Shapiro (1977), and Bellmore and Malone (1968). They all rely on the fact that there is a good algorithm for minimizing the sum $\sigma(\pi) = \Sigma_{i=1}^{n} \alpha(i,\pi(i))$ over all permutations $\pi:N\to N$ (this is a form of the assignment problem), while (5) requires minimization of $\sigma(\pi)$ over the cyclic permutations.

Some approximate algorithms for (5) that have good average-time behavior are mentioned in Section 6.

B. Shortest Paths

Shortest path algorithms were surveyed by Dreyfus (1969) and discussed in detail by Domschke (1972) and Lawler (1976). In view of these studies, the surveys by Bradley (1975) and Lawler (1978), and the bibliographies collected by Pierce (1976) and Pape (1977), we make only a few remarks here rather than attempting to summarize a large body of material.

Problem (1) is usually solved as part of

(1*) For a given node s of G, find shortest paths from s to all other nodes accessible from s.

One might think (1) could be solved significantly faster than (1*). That is true for some interesting special classes of graphs (Hadlock, 1977, 1978), but probably not for general graphs.

Several good algorithms are available for (1*), the best-known being the $O(n^2)$ procedure of Dijkstra (1959). By a simple use of priority queues, it can also be implemented in time $O(a \log n)$ (Lawler, 1978), an advantage for sufficiently sparse graphs. By a more complicated use of priority queues, Johnson (1977) shows that for each fixed positive integer k there is an implementation that runs in time $O(\min \{n^{1 + 1/k} + a, a \log n\})$. When all arc lengths are positive integers the algorithm of Wagner (1976) solves (1*) in time $O(\max \{n,a,d\})$, where d is the maximum of the lengths of shortest paths from s to other nodes.

Applying Dijkstra's procedure n times yields an $O(n^3)$ solution of

(1**) For each pair (s,t) of nodes of G, find a shortest path from s to t.

The clever but complicated approach of Fredman (1976) shows that with $O(n^3 \log\log n/\log n)$ preprocessing an $O(n^2)$ solution of (1**) is available. The preprocessing consists of compiling a table that is then used for all instances of a given size n.

The most elegant solution of (1**) is the $O(n^3)$ algorithm of Floyd (1962). Extend the definition of arc-length by setting $\alpha(i,j) = \infty$ when (i,j) is not an arc, and then proceed as follows:

```
begin
  for k ← 1 until n do
    for i ← 1 until n do
      for j ← 1 until n do
        α(i,j) ← min {α(i,j), α(i,k) + α(k,j)}
end.
```

When this computation terminates, $\alpha(i,i)$ is the length of a shortest circuit through node i, and when $i \neq j$ $\alpha(i,j)$ is the length of a shortest path from i to j. From this information it is easy to find the paths themselves.

In contrast to the shortest-path algorithms mentioned earlier, Floyd's algorithm does not require nonnegative arc lengths. It applies to an arbitrary directed network in which no circuit is of negative length. A related algorithm of Yuval (1976), using the fast multiplication of matrices

due to Strassen (1969) and based on an extension of the RAM model that permits infinite-precision real arithmetic, solves (1**) in time $O(n^{2.81})$. Klee and Larman (1978) extend Floyd's method to find shortest paths among those satisfying various sorts of restrictions. For networks in which no circuit is of negative length, Johnson (1977) and Lawler (1978) show how to take advantage of sparseness in the directed case, and Lawler (1976) shows how to use matching algorithms to find shortest paths in the undirected case.

See Section 6 for a solution of (1**) whose average-case complexity is $O(n^2 \log^2 n)$, and see Section 7 for lower bounds on the complexity of shortest-path algorithms.

C. Spanning Trees and Optimization in Independence Systems

Let us turn now to the minimum spanning tree problem (2). For a connected undirected graph $G = (N,A)$ and nonnegative arc-length function α, we seek a spanning tree of minimum weight. Call a subset of A independent if it contains no circuit. And for $x \in X \subset A$, say that x is a first member of X if $\alpha(x) \leq \alpha(x')$ for all $x' \in X$. Kruskal (1956) showed a minimum spanning tree T can be constructed as follows, where U is the set of unexplored arcs.

```
begin
  T ← Ø; U ← A;
  while U ≠ Ø do begin
                   u ← a first member of U;
                   U ← U~{u};
                   if Tu{u} is independent then T ← Tu{u}
                 end;
  print T
end.
```

Kruskal's algorithm can be implemented in time $O(a \log n)$. A later algorithm of Prim (1957) and Dijkstra (1959) solves (2) in time $O(n^2)$ or, as implemented by Kerschenbaum and Van Slyke (1972), in time $O(a \log n)$. For sparse graphs the best algorithms are those of Yao (1975) and Cheriton and Tarjan (1976), which are $O(a \log \log n)$. On the other hand, for each positive integer k there are algorithms of Johnson (1975) and Cheriton and Tarjan (1976) which, when applied to graphs having $a \geq cn^{1 + 1/k}$ for some fixed $c > 0$, solve (2) in time $O(a)$.

Actually, Kruskal's procedure finds minimum spanning trees without any restriction on the sign of α, so the same procedure can be used to find maximum spanning trees; simply redefine first member to mean of maximum α-value. The name of greedy algorithm, proposed by Edmonds (1971), is then especially appropriate, for one attempts at each stage to swallow (add to T) the largest member u of U, refraining only if u is immediately unpalatable. No attention is paid to the possibility that this greedy choice, however tempting

at the moment, may in the long run cause indigestion! And, in fact, the greedy algorithm not only solves (2) but works in the more general matroid setting described below.

The theory of matroids, which can be axiomatized in many equivalent ways, was invented by Whitney (1933) as a generalization of the theory of linear independence. Let us define an independence system as a finite family $\underset{\sim}{I}$ of finite sets, called independent sets, such that every subset of an independent set is independent. (In the example above, a set of arcs is independent if it contains no circuit.) A matroid is an independence system $\underset{\sim}{I}$ such that for each subset S of $\bigcup\underset{\sim}{I}$, all independent subsets of S are of the same cardinality. An equivalent condition is that whenever I and J are independent sets with $|I| < |J|$, there exists $j \in J$ such that $I \cup \{j\}$ is independent. A base of a matroid $\underset{\sim}{I}$ is a maximal member of $\underset{\sim}{I}$. (In the example, the bases are the spanning trees of G.) Lawler (1976) lists several books and other references on matroid theory, the latest book being that of Welsh (1976). Bruno and Weinberg (1976a,b) present matroid theory as the proper foundation for the study of electrical networks, an important advantage being that the well-developed duality theory of matroids can replace the purely graph-theoretic duality notions which are satisfactory only for planar graphs.

If $\underset{\sim}{I}$ is a matroid, $<$ is a linear ordering of $\bigcup\underset{\sim}{I}$, and "first" is interpreted in terms of $<$, then with A replaced by $\bigcup\underset{\sim}{I}$ the above algorithm produces a base T of $\underset{\sim}{I}$ that is optimal in the following strong sense: for any other base T' of $\underset{\sim}{I}$ there is a bijection $\phi: T \to T'$ with

$t \leq \phi(t)$ for all $t \in T$. This was established by Rado (1957), rediscovered independently by Gale (1968), Welsh (1968) and Edmonds (1971), and extended to infinite matroids by Klee (1971). In addition to the spanning tree problem, Lawler (1976) describes a "semimatching" problem, a job-sequencing problem, and an experimental design problem that have underlying matroid structures and hence can be solved efficiently by the greedy algorithm. Of course, the complexity of an implementation depends on the difficulty of recognizing members of $\underset{\sim}{I}$.

Each independence system $\underset{\sim}{I}$ can be expressed in various ways as the intersection of a finite number of matroids. Many important problems of combinatorial optimization can be cast in the following form: For an independence system $\underset{\sim}{I}$ that is the intersection of matroids $\underset{\sim}{I}_1,\ldots,\underset{\sim}{I}_k$ of specified sorts, and for a nonnegative function α on $\bigcup\underset{\sim}{I}$, find a member of $\underset{\sim}{I}$ that is of maximum α-weight--that is, maximize $\Sigma_{i\in I}\,\alpha(i)$ over all $I \in \underset{\sim}{I}$. Lawler (1976, Chap. 8) observes that the traveling salesman problem (5) and the 3-dimensional assignment problem (4''') can be formulated in this way for $k = 3$, so good exact algorithms should not be expected for general three-matroid optimization problems. However, for $k = 2$ Lawler (1975, 1976) has both a "primal" algorithm analogous to a procedure for finding minimum cost network flows and a "primal-dual" algorithm analogous to a procedure for finding maximum matchings. The directed analogue of the minimum spanning tree problem (2) can be handled in this way,

but is solved more efficiently by the special methods of Chu and Liu (1965) and Edmonds (1967). See also Lawler (1976) and his references, and Tarjan (1977).

When $\underset{\sim}{I}$, k and α are as in the preceding paragraph, and "first" means of maximum α-value, the greedy procedure described earlier can be used as a heuristic for finding independent sets of large weight. Results of Jenkyns (1976) and Korte and Hausmann (1978) show that if I_o is a member of $\underset{\sim}{I}$ of maximum α-weight and I_g is the output of the greedy heuristic as applied to $\underset{\sim}{I}$ and α, then

$$\frac{\Sigma_{i \in I_g} \alpha(i)}{\Sigma_{i \in I_o} \alpha(i)} \geq \min_{S \subset \mathbf{U}\underset{\sim}{I}} \frac{\ell r(S)}{ur(S)} \geq \frac{1}{k},$$

where $\ell r(S)$ ⟨resp. $ur(S)$⟩ is the minimum ⟨resp. maximum⟩ of the cardinalities of the independent subsets of S. Both inequalities are sharp in certain senses. See Korte (1978) for a fuller description of "greedy" heuristics, their uses and limitations, and for additional references.

6. THREE RECENT ADVANCES

As is clear from the recent dates of many of the cited references, the field of combinatorial optimization is in a very active stage of development. Three of the most exciting recent advances have been the Dinic-Karzanov maximum

flow algorithm, Minty's good algorithm for finding maximum independent sets in a large class of graphs, and a collection of algorithms whose average-case behavior is remarkably good.

A. The Dinic-Karzanov Maximum Flow Algorithm

The papers of Dinic (1970) and Karzanov (1974) were mentioned earlier. Like other maximum flow algorithms, the Dinic-Karzanov algorithm starts with a feasible flow ϕ, uses ϕ and the underlying network to produce an auxiliary network, and by a computation in the latter attempts to find an improvement ϕ' of ϕ. If an improvement ϕ' is found, the computation is repeated with ϕ' in the role of ϕ, and when no improvement is found then ϕ is optimal. However, in the new algorithm the improvements are not made by flow-augmenting paths but in a more global manner that leads to an $O(n^3)$ algorithm. The announcement of Karzanov (1974) outlines the method but does not include details of proof. Details can be found in the book of Adelson-Velsky et al. (1975) (in Russian) and in the exposition of Even (1976). The algorithm has been improved to $O(n^{5/3} a^{2/3})$ by Z. Galil.

B. Minty's Algorithm for Maximum Independent Sets

The maximum independent set problem (4'') is NP-complete, even when restricted to 3-valent planar graphs (Garey, Johnson and Stockmeyer, 1976), but one may hope for good algorithms that solve the problem for special classes of graphs. For references to such algorithms, and to other restrictions under which the problem remains NP-complete, see Garey and Johnson (1978?). In the bipartite case, (4'') is well-solved by matching (Lawler, 1976, pp. 189-196). By far the most successful attack on the nonbipartite case is due to Minty (1978?), who shows that matching techniques are also applicable to all graphs that are claw-free. A claw is a quadruple of nodes (v,w,x,y) such that each node in the set $S = \{w,x,y\}$ is adjacent to v but no two members of S are adjacent.

To appreciate how close Minty's result comes to the boundary of NP-completeness, consider the restricted form of the 3-dimensional assignment problem mentioned at the end of Section 4. Let G be the graph whose nodes are the ordered triples (i,j,k) of integers between 1 and n for which $\alpha_{ijk} = 1$, two nodes being adjacent if they agree in at least one coordinate. Then G may have claws but it is free of quintuples (v,w,x,y,z) such that each node in the set $S = \{w,x,y,z\}$ is adjacent to v but no two members of S

are adjacent. The problem of deciding, for each such G, whether G contains an independent set of cardinality n, is NP-complete.

Minty also considers the weighted version of (4''), in which a node-weighting α is given and one seeks an independent set of maximum weight. He shows that for claw-free graphs this problem is reducible to the weighted matching problem and hence solvable in polynomial time.

C. Some Algorithms with Good Average-Case Behavior

From the worst-case viewpoint, some well-solved problems of combinatorial optimization have been so intensively studied that further significant improvement in solution methods seems unlikely; also, most researchers doubt the existence of good algorithms for the NP-complete problems. However, neither of these comments applies to the average-case viewpoint. This fact, along with the belief that average-case behavior is more important than worst-case behavior in many applications, has motivated the search for algorithms of good average-case behavior.

Spira (1973) and Karp (1978) consider a complete directed network $((N,A),\alpha)$ whose arc lengths $\alpha(i,j)$ are all drawn independently from the same probability distribution in $]0,\infty[$. Spira proposes an algorithm for the all-pairs shortest path problem (1**) and shows that for any distribution the expected running time is $O(n^2 \log^2 n)$. Fredman (1976)

shows how to reduce the expected number of comparisons (but not the overall complexity) to $O(n^2 \log n)$. Carson and Law (1977) report on computations comparing Spira's method for various distributions with an improvement of the method of Dijkstra (1959) due to Yen (1972).

Karp (1978) proposes an $O(n^3)$ approximate algorithm for the traveling salesman problem (5). His algorithm first solves the $n \times n$ assignment problem for the matrix $(\alpha(i,j))$ and then patches together the cycles of the resulting permutation to form a tour. He shows that if the $\alpha(i,j)$ are drawn independently from the uniform distribution on $]0,1[$ then, with probability tending to 1 as $n \to \infty$, the ratio of the length of the tour produced by the algorithm to that of an optimum tour is $< 1 + \varepsilon(n)$, where $\varepsilon(n) \to 0$ as $n \to \infty$. A tool in his proof is a result of Walkup (1977) on the expected value of a random assignment problem. Similar patching schemes appear in algorithms proposed by Karp (1977) for the traveling salesman problem in the Euclidean plane.

In the results of Spira and Karp stated above, the underlying graph is always complete; only the arc lengths vary at random. For other results on the average-case behavior of algorithms, a notion of a random undirected graph that has node-set $N = \{1,\ldots n\}$ and has a specified density p (or has a specified number E of arcs) is required. Let U_n denote the set of all $n(n-1)/2$ unordered pairs of distinct integers between 1 and n, each pair representing a possible arc of a graph with node-set N. In the "p" model, a sample graph is formed by including each member of U_n with probability p, independent of what other arcs are

chosen. In the "E" model, the sample space consists of all $A \subset U_n$ for which $|A| = E$, all such arc-sets A being drawn with equal probabilities. When $E = pn(n-1)/2$, the two models are closely related (Erdös and Rényi, 1961); Angluin and Valiant (1977) show that if in a certain sense an algorithm works for one model then it works for the other one also.

A matching is perfect if it covers all nodes. Erdös and Rényi (1966) show that if $E(n) = \frac{1}{2}n \log n + \omega(n)n$ and if $PM(n)$ is the probability that a random graph with n nodes and $E(n)$ arcs admits a perfect matching, then as (even) $n \to \infty$, $PM(n) \to 1$ if $\omega(n) \to \infty$. Using a model of computation similar to but more flexible than the RAM model, Angluin and Valiant (1977) show there exists an algorithm that has the following properties:

(a) accepting as input an undirected graph G with n nodes, it outputs a perfect matching or concludes correctly that G does not admit one;

(b) there exist positive constants c and k such that for each n and for each $E \geq cn \log n$, the expected run-time is less than $kn \log n$ for random graphs with n nodes and E arcs.

Pósa (1976) shows that if $E(n) = \beta n \log n$ with $\beta > 1$, and if $T(n)$ is the probability that a random graph with n nodes and $E(n)$ arcs admits a tour, then $T(n) \to 1$ as $n \to \infty$. Karp (1976) uses Posa's argument as the basis of a polynomial time algorithm for finding tours almost surely.

Considering spanning simple paths as well as tours, Angluin and Valiant (1977) describe an algorithm that has the following properties:

(a) accepting as input an undirected graph G with n nodes, and a pair of nodes s, t with $1 \le s \le t \le n$, it attempts to find a simple path from s to t that uses all nodes (a tour when $s = t$);

(b) for each $\gamma > 0$ there exist positive constants c and k such that for each n, for each $E \ge cn \log n$, and for each pair s, t, the probability is at least $1 - O(n^{-\gamma})$ (for random graphs with n nodes and E arcs) that the desired path is found in fewer than $kn \log n$ steps.

Essentially the same results are obtained by Angluin and Valiant (1977) for the problem of finding a tour or a spanning simple path in a directed graph. Their complexity estimates are based on a model of computation in which randomized decisions are possible, but are worsened only by an additional factor of $\log n$ when that possibility is removed from the model.

See Karp (1976) for the results of probabilistic analyses of other combinatorial search algorithms, including use of the greedy algorithm to find large independent sets of nodes. And see Evans (1976) for a treatment of maximum flow problems in probabilistic graphs. Also, see the last two paragraphs of Section 7.

7. THREE DIRECTIONS FOR RESEARCH

Since the field of combinatorial optimization is developing so rapidly, it provides many directions for research and many specific problems that beg for solution. Rather than discussing individual problems, we focus on the three directions of research that seem respectively to be most promising for

(a) elucidation of the fundamental theoretical issues;

(b) unification of the existing theory;

(c) production of useful or interesting algorithms.

A. *Finding Lower Bounds*

The goal here is simply stated: Find sharp lower bounds for the computational complexity of the important problems of combinatorial optimization, preferably with respect to the RAM or some closely related model of random access computation. When a problem deals with networks $((N,A),a)$, we say the problem is of complexity $\Omega(\tau(n,a))$ if there is a constant $c > 0$ such that every algorithm for solving the problem requires at least $c\tau(n,a)$ steps for worst-case input with parameters n and a. This lower bound is <u>sharp</u> if the problem can be solved by an algorithm of complexity $O(\tau(n,a))$, so that the worst-case behavior of the problem is determined to within a multiplicative constant.

At least with respect to the RAM model, this direction of research appears to be very difficult. In particular, it involves deciding whether $\underset{\sim}{P} = \underset{\sim}{NP}$. Nevertheless, it is described as "promising" because it deals with such important questions. Until sharp lower bounds are found for a variety of problems of combinatorial optimization, there will be no clear understanding of why some problems are easy and others superficially resembling them are very hard. An upper bound on the intrinsic complexity of a problem can be established by designing and analyzing an algorithm for its solution. If the design involves some ingenuity and the analysis is done with care, the result is likely to be interesting and perhaps even useful. At the very least, it is likely to be "non-trivial" in an intuitive sense. Nontrivial lower bounds are much harder to come by, because they require consideration of all conceivable algorithms and thus demand a much clearer understanding of the underlying logical issues. In fact, as Weide (1977) remarks, trivial lower bounds are often the only ones available. Consider, for example, the problem of finding a longest simple path between two given nodes of a complete undirected network with n nodes. The problem is $\underset{\sim}{NP}$-hard and hence probably does not admit a good algorithm --not even one of complexity $O(n^{1000^{1000}})$. However, the best known lower bound (with respect to the RAM model) is the trivial one of $\Omega(n^2)$, obtained by noting that none of the $n(n-1)/2$ arc lengths can be ignored by any algorithm that solves the problem for all sets of input data.

Weide (1977) discusses a few methods that have been used to establish lower bounds, with respect to various models of computation, on the complexity of combinatorial problems. The interested reader should consult Kerr (1970), Spira and Pan (1975), and Yao et al. (1977) concerning shortest paths, Spira and Pan (1975) and Shamos and Hoey (1975) concerning minimum spanning trees, Harper and Savage (1972) concerning maximum matching, Harper and Savage (1972) and Rabin (1972) concerning shortest tours, Holt and Reingold (1972) concerning the detection of circuits or connectedness, and Rivest and Vuillemin (1975) concerning the detection of an artitrary nontrivial monotone graph property (a property possessed by some but not all graphs and preserved by addition of arcs). The most promising method seems to me to be the one of Rabin (1972), which is used also by Spira and Pan (1974) and Yao et al. (1977). (The application to shortest paths by Yao et al. (1977) is apparently invalid because it depends on an incorrect counting argument, but the paper is still of interest.) Some other papers related to Rabin's method are those of Spira (1972), Klee (1975) and Yao (1976).

B. Optimization in Oriented Matroids

The theory of matroids is a combinatorial generalization of the theory of linear dependence in a vector space over a field, and of the theory of linear equalities. Similarly, the theory of oriented matroids may be viewed as a combinatorial generalization of the theory of positive linear

dependence in a vector space over an ordered field, and of the theory of inequalities. The notion of an oriented matroid developed from the digraphoids of Minty (1966) through the generalizations of Camion (1968), Fulkerson (1968) and Rockafellar (1969), to the present axiomatic versions of Bland (1974), Las Vergnas (1975) and Lawrence (1975). A good basic reference is Bland and Las Vergnas (1978).

Realizing a possibility first suggested by Rockafellar (1969), Bland (1974, 1976) and Lawrence (1975) show the basic parts of linear programming theory can be extended to oriented matroids. Both have oriented matroid formulations of linear programming duality, and Bland (1976, 1977) also has a finite pivoting method which specializes, in the context of linear programming, to Dantzig's simplex method with a new pivot selection rule. The study of optimization in the framework of oriented matroids seems to offer the best chance for the unification of large parts of linear and combinatorial optimization. This is related to the approach of Lawler (1976, Chap. 9) through matroids with parity conditions, and it is important to clarify the relationships between the two approaches.

C. Development of Algorithms

A striking aspect of combinatorial mathematics, and of combinatorial optimization in particular, is its endless supply of interesting problems. To become convinced that the design and analysis of algorithms for combinatorial optimiza-

tion will continue for a long time, one need only glance at a few volumes of Annals of Discrete Mathematics, Information Processing Letters, Journal of the Association for Computing Machinery, Mathematical Programming, Mathematical Programming Studies, Networks, Operations Research, Proceedings of ACM Symposia on the Theory of Computing, Proceedings of IEEE Symposia on the Foundations of Computer Science, or the SIAM Journal on Computing. Below are described some directions of algorithm development that seem at present to be especially promising.

Exploring the boundary of NP-completeness

By following the advice in the excellent chapter of Garey and Johnson (1978?) on using NP-completeness to analyze problems, designers of algorithms can help to "explore the boundary" of NPC, the class of all NP-complete problems. The general idea is that, having established the membership of a problem in one of the classes P and NPC, one should not rest before finding a closely related problem in the other class. For each of the problems (1) - (4), Garey and Johnson (1978?) describe several relatives in NPC, and for (5) there are relatives in P. The reader has met some of these relatives earlier, and a few more are described below.

The Chinese postman is a relative of the traveling salesman. In serving his route he must traverse each street at least once, one-way streets only in the proper direction, and

return to his starting point. And of course he wants to minimize the distance traveled. The postman's problem is in $\underset{\sim}{P}$ if all streets are directed or all are undirected (Edmonds and Johnson, 1973), but the mixed case is $\underset{\sim}{\underset{\sim}{NP}}$-complete (Papadimitriou, 1976).

When $G = (N,A)$ and $H = (V,E)$ are undirected graphs, a perfect H-packing of G is a partition of N into pairwise disjoint subsets each of which is of cardinality $|V|$ and induces in G a subgraph isomorphic to H. For fixed H, the problem of deciding whether G admits a perfect H-packing belongs to $\underset{\sim}{P}$ if $|V| \leq 2$ (use matching) but is $\underset{\sim}{\underset{\sim}{NP}}$-complete whenever $|V| \geq 3$ (Kirkpatrick and Hell, 1978).

Let $\underset{\sim}{\underset{\sim}{NPI}} = \underset{\sim}{\underset{\sim}{NP}} \sim (\underset{\sim}{P} \cup \underset{\sim}{\underset{\sim}{NPC}})$, the class of all problems in $\underset{\sim}{\underset{\sim}{NP}}$ that are of "intermediate" difficulty in the sense that, even though they can't be solved by polynomially bounded algorithms, they're at least not so difficult as to be $\underset{\sim}{\underset{\sim}{NP}}$-complete. Under the assumption that $\underset{\sim}{P} \neq \underset{\sim}{\underset{\sim}{NP}}$, Ladner (1975) proves $\underset{\sim}{\underset{\sim}{NPI}}$ is nonempty but no one has produced a member of $\underset{\sim}{\underset{\sim}{NPI}}$ that is of practical importance or intrinsic interest aside from its membership in $\underset{\sim}{\underset{\sim}{NPI}}$. As noted by Garey and Johnson (1978?), linear programming is a prime candidate. A class of problems that seems well suited to exploring the boundary of $\underset{\sim}{\underset{\sim}{NPC}}$, and perhaps of $\underset{\sim}{\underset{\sim}{NPI}}$, is obtained as follows. Let F be the set of all functions $f: Z_o^+ \to Z_o^+$ such that $f(k) \leq k$ for all $k \in Z_o^+$, the set of all nonnegative integers. For each $f \in F$ an instance of the problem Q_f ⟨resp. R_f⟩ is associated with a graph

$G = (N,A)$, a pair (x,y) of distinct members of N, and a sequence S of $f(n-2)$ distinct member of $N\sim\{x,y\}$. The problem Q_f ⟨resp. R_f⟩ asks whether there is a simple path from x to y that uses all the members of S in some ⟨resp. in the specified⟩ order, other intermediate nodes being permitted as well. Note that Q_f belongs to $\underset{\sim}{NPC}$ when $f(k) \equiv k$, while R_f belongs to $\underset{\sim}{P}$ whenever $k - f(k)$ is bounded. When $f \equiv 0$, both problems belong to $\underset{\sim}{P}$. When $f \equiv 1$ and G is undirected, both problems belong to $\underset{\sim}{P}$. (Add a new node t, new arcs $\{t,x\}$ and $\{t,y\}$. Let all arcs have capacity 1. Solve as a maximum flow problem with sink t and source s, where $S = \{s\}$.) What happens for other choices of f?

Approximate algorithms

We saw in the result of Sahni and Gonzalez (1976) that unless $\underset{\sim}{P} = \underset{\sim}{NP}$, the general undirected traveling salesman problem (5) does not admit a polynomially bounded algorithm that is ε-approximate, no matter how large ε is chosen. On the other hand, when G's arc lengths satisfy the triangle inequality, the $O(n^3)$ method of Christofides (1976) is $\frac{1}{2}$-approximate and it may even turn out that for each $\varepsilon > 0$ this special case of (5) admits a polynomially bounded ε-approximate algorithm. In short, though the $\underset{\sim}{NP}$-complete problems are computationally equivalent so far as solvability by polynomially bounded exact algorithms is

concerned, this does not apply to approximate algorithms. Much is known about approximate algorithms but much more remains to be discovered. Good sources of information are the annotated bibliography of Garey and Johnson (1976), the survey article of Korte (1978), and the book of Garey and Johnson (1978?).

Average-case and probabilistic analysis of algorithms

In Part C of Section 6, three sorts of algorithms are lumped together as "algorithms with good average-case behavior." Some are precisely that, while others are with high probability $\varepsilon(n)$-approximate, where $\varepsilon(n) \to 0$ as $n \to \infty$. A period of considerable activity in the average-case and probabilistic analysis of algorithms for combinatorial optimization is now under way. A basic paper is Karp (1976), and Weide (1977) also has a brief introduction to the subject.

The papers on random graphs by Erdös and Rényi do not treat algorithms explicitly, but they contain many facts that are essential for the designer of algorithms in this area; several of these papers are collected in Erdös (1973). In addition to these and the papers mentioned in Part C of Section 6, the interested reader should consult Grimmett and McDiarimid (1975), Walkup (1977b), Lueker (1978), Karp (1978?) Cohen et al. (1978?), and Klee and Larman (1978?a,b).

8. WHAT ARE THE BEST SOURCES OF FURTHER INFORMATION?

As is clear from our frequent reference to them, the two indispensable books for the study of combinatorial optimization are those of Lawler (1976) for problems in $\underset{\sim}{P}$ and of Garey and Johnson (1978?) for problems in $\underset{\sim}{NPC}$. But even they cannot cover everything in detail, and many of the references cited by them or in our list of references are also necessary for a thorough understanding of the field.

We close by mentioning two very active areas of combinatorial optimization whose omission here causes regret. Though both are very important for the solution of practical problems, they have been omitted because of space considerations and because of our decision to base the entire exposition on problems (1) - (5). For access to the tremendous volume of material on Scheduling (several books, hundreds of research papers), a good starting point is the special issue of Operations Research devoted to the subject. It is listed here under Florian (1978).

For discrete location problems, see Krarup and Pruzan (1978) and their references.

REFERENCES

In the list of references below, a question mark after a date indicates that the paper in question had not yet appeared when this report was completed in May of 1978.

Adelson-Velsky, G.M., Dinic, E.A., and Karzanov, A.V. (1975). "Flow Algorithms" (in Russian). Nauka, Moscow.

Agin, N. (1966). Optimum seeking with branch-and-bound. Management Sci. 13:176-185.

Aho, A.V., Hopcroft, J.E., and Ullman, J.D. (1976). "The Design and Analysis of Computer Algorithms." Addison-Wesley, Reading, Mass..

Anderson, W.N. (1975). Maximum matching and the rank of a matrix. SIAM J. Appl. Math. 28:114-123.

Angluin, D. and Valiant, L.G. (1977). Fast probabilistic algorithms for Hamiltonian circuits and matchings. Proc. ACM Symp. Theory of Computing 9:30-41.

Avis, D. and Chvátal, V. (1976). Notes on Bland's pivoting rule. Tech. Report, Univ. of Montreal.

Balinski, M.L. (1974), editor. "Approaches to Integer Programming." Math. Programming Studies, vol. 1. North-Holland Pub. Co., Amsterdam.

Balinski, M.L., and Hoffman, A. (1978?), editors. "Polyhedral Combinatorics." Math. Programming Studies, vol. ?. North-Holland Pub. Co., Amsterdam.

Busacker, R.G., and Saaty, T.L. (1964). "Finite Graphs and Networks." Mc-Graw-Hill, New York.

Camion, P. (1968). Modules unimodulaires. J. Combinatorial Theory 4:301-362.

Carson, J.S., and Law, A.M. (1977). A note on Spira's algorithm for the all-pairs shortest-path problem. SIAM J. Comput. 6:696-699.

Cheriton, D., and Tarjan, R.E. (1976). Finding minimum spanning trees. SIAM J. Comput. 5:724-742.

Christofides, N. (1976). Worst-case analysis of a new heuristic for the traveling salesman problem. Tech. Report, Grad. School Industrial Admin., Carnegie-Mellon Univ. Abstract in "Algorithms and Complexity--New Directions and Recent Results." (J.F. Traub, ed.), p. 441. Academic Press, New York.

Chu, Y. and Liu, T. (1965). On the shortest arborescence of a directed graph. Scientia Sinica (Peking) 4:1396-1400.

Chvátal, V. (1973). Edmonds polytopes and a hierarchy of combinatorial problems. Discrete Math. 4:305-337.

Chvátal, V. (1975). On certain polytopes associated with graphs. J. Combinatorial Theory B 18:138-154.

Chvátal, V. (1976). D. Ray Fulkerson's contributions to operations research. Math. Operations Research. 1:311-320.

Cobham, A. (1965). The intrinsic computational difficulty of functions. Proc. 1964 International Congress for Logic, Methodology and Philosophy of Science (Y. Bar-Hillel, ed.), pp. 24-30. North-Holland Pub. Co., Amsterdam.

Bellmore, M. and Malone, J.C. (1971). Pathology of traveling salesman subtour-elimination algorithms. Operations Res. 19:278-307.

Bellmore, M., and Nemhauser, G. (1968). The traveling salesman problem: A survey. Operations Res. 16:538-558.

Berge, C. (1957). Two theorems in graph theory. Proc. Nat. Acad. Sci. U.S.A. 43:842-844.

Berge, C., and Ghouila-Houri, A. (1965). "Programming, Games and Transportation Networks." Wiley, New York.

Billera, L.J., and Lucas, W,F, (1976). Delbert Ray Fulkerson. Math. Operations Res. 1:299-310.

Bland, R.G. (1974). Complementary orthogonal subspaces of R^n and orientability of matroids. Ph.D. dissertation, Operations Res. Dept., Cornell Univ.

Bland, R.G. (1976). A combinatorial abstraction of linear programming. *CORE Discussion Paper 7631. Center for* Operations Research and Econometrics, Heverlee, Belgium.

Bland, R.G. (1977). New finite pivoting rules for the simplex method. Math. Operations Res. 2:103-107.

Bland, R.G., and Las Vergnas, M. (1978). Orientability of matroids. J. Combinatorial Theory B 24:94-123.

Bradley, G.H. (1975). Survey of deterministic networks. Trans. Amer. Inst. Industrial Eng. 7:222-234.

Bruno, J., and Weinberg, L. (1976a). Generalized networks: Networks embedded on a matroid, Part I. Networks 6:53-94.

Bruno, J., and Weinberg, L. (1976b). Generalized networks: Networks embedded on a matroid, Part II. Networks 6: 231-272.

Cohen, J.E., Komlós, J., and Mueller, T. (1978?). The probability of an interval graph, and why it matters. "Relations between Combinatorics and other parts of Mathematics." Amer. Math. Soc. Proc. Symp. Pure Math.

Cook, S.A. (1971). The complexity of theorem-proving procedures. Proc. ACM Symp. Theory of Computing 3:151-158.

Cook, S.A. and Reckhow, R.A. (1973). Time-bounded random access machines. J. Comput. System Sci. 7:354-375.

Cornuejols, G. and Nemhauser, G.L. (1978). Tight bounds for Christofides' traveling salesman heuristic. Math. Programming 14:116-121.

Dantzig, G.B. (1963). "Linear Programming and Extensions." Princeton Univ. Press.

Dantzig, G.B., Ford, L.R.Jr., and Fulkerson (1956). A primal-dual algorithm for linear programs. "Linear Inequalities and Related Systems" (H.W. Kuhn and A.W. Tucker, eds.). Ann. Math. Studies 38:171-181.

Dantzig, G.B., Fulkerson, D.R., and Johnson, S. (1959). On a linear programming combinatorial approach to the traveling salesman problem. Operations Res. 7:58-66.

Desler, J.F., and Hakimi, S.L. (1969). A graph-theoretic approach to a class of integer-programming problems. Operations Res. 17:1017-1033.

Dijkstra, E.W. (1959). A note on two problems in connexion with graphs. Numer. Math. 1:269-271.

Dinic, E.A. (1970). Algorithm for solution of a problem of maximum flows in a network with power estimation. Soviet Math. Dokl. 11:1277-1280.

Domschke, W. (1972). "Kürzeste Wege in Graphen: Algorithmen, Verfahrensvergleiche." Mathematical Systems in Economics, vol. 2. Verlag Anton Hain, Meisenheim am Glan.

Dreyfus, S.E. (1969). An appraisal of some shortest-path algorithms. Operations Res. 17:395-412.

Eastman, W.L. (1958). Linear programming with pattern constraints. Ph.D. dissertation, Harvard Univ.

Edmonds, J. (1965a). Paths, trees and flowers. Canad. J. Math. 17:449-467.

Edmonds, J. (1965b). Maximum matchings and a polyhedron with 0-1 vertices. J. Res. Nat. Bur. Standards Sect. B 69:125-130.

Edmonds, J. (1967). Optimum branchings. J. Res. Nat. Bur. Standards Sect. B 71:233-240. Also in "Mathematics of the Decision Sciences, Part I" (G. B. Dantzig and A. F. Veinott, eds.). Amer. Math. Soc. Lect. Appl. Math. 11:335-345.

Edmonds, J. (1970). Submodular functions, matroids and certain polyhedra. In "Combinatorial Structures and their Applications." Proc. Calgary International Conference (R. Guy, ed.), pp. 69-87. Gordon and Breach, New York.

Edmonds, J. (1971). Matroids and the greedy algorithm. Math. Programming 1:127-136.

Edmonds, J., and Johnson, E.L. (1973). Matching, Euler tours, and the Chinese postman. Math. Programming 5:88-124.

Edmonds, J., and Karp, R.M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. J. ACM 19:248-264.

Edmonds, J., and Pulleyblank, W. (1978?). "Optimum Matchings." Johns Hopkins Univ. Press.

Eiselt, H.A., and von Frajer, H. (1977). "Operations Research Handbook--Standard Algorithms and Methods with Examples." W. de Gruyter, Berlin-New York.

Elias, P., Feinstein, A., and Shannon, C.E. (1956). Note on maximum flow through a network. IRE Trans. Information Theory IT-2:117-119.

Elmaghraby, S. (1970). "Some Network Models in Operations Research." Springer, New York.

Erdös, P. (1973). "The Art of Counting--Selected Writings" (J. Spencer, ed.). M.I.T. Press.

Erdös, P., and Rényi, A. (1961). On the evolution of random graphs. Bull. Inst. Internat. Statistics 38:343-347.

Erdös, P., and Rényi, A. (1966). On the existence of a factor of degree one of connected random graphs. Acta Math. Acad. Sci. Hungar. 17:359-379.

Evans, J.R. (1976). Maximum flow in probabilistic graphs--the discrete case. Networks 6:161-183.

Even, S. (1976). The max flow algorithm of Dinic and Karzanov: an exposition. Report MIT/LCS/TM-80, Laboratory for Computer Science. Mass. Inst. Tech.

Even, S. and Kariv, O. (1975). An $O(N^{2.5})$ algorithm for maximum matching in general graphs. Proc. IEEE Symp. Foundations of Computer Science 16:100-112.

Feicht, E., Heck, J. and U. Pape (1977). Algorithmen von Edmonds und Gabow zur Ermittlung des maximalen Matchings in schlichten und endlichen Graphen. Bericht Nr. 75-15, Fachbereich Informatik. Technische Universität, Berlin.

Florian, M., Maxwell, W., and Ratliff, H.D. (1978), editors. "Scheduling." Vol. 26, No. 1 of Operations Res.

Floyd, R.W. (1962). Algorithm 97, Shortest path. Comm. ACM 5:345.

Ford, L.R.Jr., and Fulkerson, D.R. (1956). Maximal flows through a network. Canad. J. Math. 8:399-404.

Ford, L.R.Jr., and D.R. Fulkerson (1957). A simple algorithm for finding maximal network flows and an application to the Hitchcock problem. Canad. J. Math. 9:210-218.

Ford, L.R.Jr., and Fulkerson, D.R. (1962). "Flows in Networks." Princeton Univ. Press.

Frank, H. and Frisch, I. (1972). "Communication, Transportation and Flow Networks." Addison-Wesley, Reading, Mass.

Fredman, M.L. (1975). On the decision tree complexity of the shortest path problems. Proc. IEEE Symp. Foundations of Computer Science 16:98-99.

Fredman, M.L. (1976). New bounds on the complexity of the shortest path problem. SIAM J. Comput. 5:83-89.

Fulkerson, D.R. (1966). Flow networks and combinatorial operations reserach. Amer. Math. Monthly. 73:115-138.

Fulkerson, D.R. (1968). Networks, frames, blocking systems. "Mathematics of the Decision Sciences, Part I" (G.B. Dantzig and A.F. Veinott, eds.). Amer. Math. Soc. Lect. Appl. Math. 11:303-334.

Fulkerson, D.R. (1971). Blocking and antiblocking pairs of polyhedra. Math. Programming 1:168-194.

Fulkerson, D.R. (1973). On the perfect graph theorem. In "Mathematical Programming" (T.C. Hu and S.M. Robinson, eds.), pp. 69-77. Academic Press, New York.

Gabow, H.N. (1973). Implementations of algorithms for maximum matching on nonbipartite graphs. Ph.D. dissertation, Computer Sci. Dept., Stanford Univ.

Gabow, H.N. (1976). An efficient implementation of Edmonds' algorithm for maximum matching on graphs. J. ACM 23:221-234.

Gale, D. (1968). Optimal assignments in an ordered set: an application of matroid theory. J. Combinatorial Theory 4:176-180.

Garey, M.R., Graham, R.L. and Johnson, D.S. (1976). Some NP-complete geometric problems. Proc. ACM Symp. Theory of Computing 8:10-22.

Garey, M.R., and Johnson, D.S. (1976). Approximation algorithms for combinatorial problems: an annotated bibliography. In "Algorithms and Complexity--New directions and Recent Results" (J.F. Traub, ed.), pp. 41-52. Academic Press, New York.

Garey, M.R., and Johnson, D.S. (1978?). "Computers and Intractability: A Guide to the Theory of NP-Completeness." Freeman, San Francisco.

Garey, M.R., Johnson, D.S., and Stockmeyer, L. (1976). Some simplified NP-complete graph problems. Theoretical Comput. Sci. 1:237-267.

Garey, M.R., Johnson, D.S., and Tarjan, R.E. (1976). The planar Hamiltonian circuit problem is NP-complete. SIAM J. Comput. 5:704-714.

Garfinkel, R.S. (1977). Minimizing wallpaper waste, Part I: A class of traveling salesman problems. Operations Res. 25:741-751.

Garfinkel, R.S. (1978). Branch and bound methods for integer programming. Working paper no. 46. College of Business Admin., Univ. of Tennessee.

Garfinkel, R.S., and Nemhauser, G.L. (1972). "Integer Programming." Wiley, New York.

Geoffrion, A.M., and Marsten, R.E. (1972). Integer programming algorithms: a framework and state-of-the-art survey. Management Sci. 18:465-491.

Gilmore, P.C. and Gomory, R.E. (1964). Sequencing a one state-variable machine: a solvable case of the traveling salesman problem. Operations Res. 12:655-679.

Golden, B.L. (1977). A statistical approach to the TSP. Networks 7:209-225.

Gomory, R.E. (1958). Outline of an algorithm for integer solutions to linear programs. Bull. Amer. Math. Soc. 64:275-278.

Gomory, R.E. (1966). The traveling salesman problem. Proc. IBM Sci. Comput. Symp. Combinatorial Problems (Yorktown Heights, N.Y., 1964), pp. 93-121. IBM Data Process. Div., White Plains, N.Y.

Grimmett, G.R., and McDiarmid, C.J.H. (1975). On colouring random graphs. Math. Proc. Cambridge Phil. Soc. 77:313-324.

Hadlock, F.O. (1977). A shortest path algorithm for grid graphs. Networks 7:323-334.

Hadlock, F.O. (1978). Optimum paths in lattice graphs. Proc. Workshop on the Information Linkage between Applied Mathematics and Industry (Monterey, 1978) (P. Wang, ed.). Academic Press, New York.

Hammer, P.L., Johnson, E.L., Korte, B.H., and Nemhauser, G.L. (1977), editors. "Studies in Integer Programming." Ann. Discrete Math. 1.

Harper, L.H., and Savage, J.E. (1972). On the complexity of the marriage problem. Advances in Math. 9:299-312.

Helbig Hansen, K., and Krarup, K. (1974). Improvements of the Held-Karp algorithm for the symmetric traveling-salesman problem. Math. Programming 7:87-96.

Held, M., and Karp, R.M. (1970). The traveling salesman problem and minimum spanning trees. Operations Res. 18:1138-1162.

Held, M., and Karp, R.M. (1971). The traveling-salesman problem and minimum spanning trees: Part II. Math. Programming 1:6-25.

Held, M., Wolfe, P., and Crowder, H.P. (1974). Validation of subgradient optimization. Math. Programming 6:62-88.

Holt, R.C., and Reingold, E.M. (1972). On the time required to detect cycles and connectivity in graphs. Math. Systems Theory 6:103-106.

Hopcroft, J.E., and Karp, R.M. (1973). An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. 2 (1973) 225-231.

Hu, T.C. (1969). "Integer Programming and Network Flows." Addison-Wesley, Reading, Mass.

Iri, M. (1969). "Network Flows, Transportation and Scheduling." Academic Press, New York.

Jenkyns, T.A. (1976). The efficacy of the "greedy" algorithm. Proc. S. E. Conf. Combinatorics, Graph Theory and Computing 7:341-350.

Jeroslow, R. (1973). The simplex algorithm with the pivot rule of maximizing criterion improvement. Discrete Math. 4:367-377.

Johnson, D.B. (1975). Priority queues with update and finding minimum spanning trees. Information Processing Letters. 4:53-57.

Johnson, D.B. (1977). Shortest paths in dense networks. J. ACM 24:1-13.

Kameda, T. and Munro, I. (1974). A $O(|V| \cdot |E|)$ algorithm for maximum matching of graphs. Computing 12:91-98.

Karg, R.L. and Thompson, G.L. (1964). A heuristic approach to solving traveling salesman problems. Management Sci. 10:225-248.

Kariv, O. (1976). An $O(n^{2.5})$ algorithm for finding a maximum matching in a general graph. Ph.D. Dissertation, Dept. of Applied Math., Weizmann Inst. Sci., Rehobot, Israel.

Karp, R.M. (1972). Reducibility among combinatorial problems. In "Complexity of Computer Computations" (R.E. Miller and J.W. Thatcher, eds.), pp. 85-103. Plenum Press, New York.

Karp, R.M. (1975a) On the computational complexity of combinatorial problems. Networks 5:45-68.

Karp, R.M. (1975b). Computational complexity of combinatorial and graph-theoretic problems. In "Theoretical Computer Science," Centro Internazionale Matematico Estivo, lectures held in Bressanove June 1975, pp. 98-184. Edizioni Cremonese, Rome.

Karp, R.M. (1976) The probabilistic analysis of some combinatorial search algorithms. In "Algorithms and Complexity--New Directions and Recent Results" (J.F. Traub, ed.), pp. 1-19. Academic Press, New York.

Karp, R.M. (1977). Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane. Math. Operations Res. 2:209-224.

Karp. R.M. (1978). A patching algorithm for the nonsymmetric traveling salesman problem. Memorandum No. UCB/ERL M78/2, Electronics Res. Lab, College of Engineering, University of California, Berkeley.

Karp, R.M. (1978?) Some connections between the theory of random graphs and the analysis of graph-theoretic algorithms. "Relations between Combinatorics and Other Parts of Mathematics." Amer. Math. Soc. Proc. Symp. Pure Math.

Karzanov, A.V., Determinizing the maximal flow in a network by the method of preflows. Soviet Math. Dokl. 15:434-437.

Kerr, L.R. (1970). The effect of algebraic structure on the computational complexity of matrix computations. Ph.D. dissertation, Computer Sci. Dept., Cornell Univ.

Kerschenbaum, A. and Van Slyke, R. (1972). Computing minimum spanning trees efficiently. Proc. 25th Ann. Conf. ACM (Boston, August 1972), vol. 1, pp. 518-527.

Kinariwala, B., and Rao, A.G. (1977). Flow switching approach to the maximum flow problem. J. ACM 24:630-645.

Kirkpatrick, D.G., and Hell, P. (1978). On the complexity of a generalized matching problem. Proc. ACM Symp. Theory of Computing 10:240-245.

Klee, V. (1971). The greedy algorithm for finitary and cofinitary matroids. "Combinatorics" (T. Motzkin, ed.). Amer. Math. Soc. Proc. Symp. Pure Math. 19:137-152.

Klee, V. (1975). Spira's theorem on complete linear proofs of systems of linear inequalities. Mathematika. 22:112-121.

Klee, V., and Larman, D. (1978). Use of Floyd's algorithm to find shortest restricted path. "Discrete Optimization" (P. Hammer et al., eds.). Ann. Discrete Math. 4:

Klee, V., and Larman, D. (1978?a). Random bipartite graphs. In preparation.

Klee, V., and Larman, D. (1978?b). Diameters of random graphs. In preparation.

Klee, V., and Minty, G.J. (1972). How good is the simplex algorithm? In "Inequalities III" (O. Shisha, ed.), pp. 159-175. Academic Press, New York.

Klee, V., and Van den Driessche, P. (1977). Linear algorithms for testing the sign stability of a matrix and for finding Z-maximum matchings in acyclic graphs. Numer. Math. 28:273-285.

Korte, B. (1978). Approximative algorithms for discrete optimization problems. "Discrete Optimization" (P. Hammer et al., eds.). Ann. Discrete Math. 4:

Korte, B., and Hausmann, D. (1978). An analysis of the greedy heuristic for independence systems. "Algorithmic Aspects of Combinatorics" (B. Alspach et al., eds.). Ann. Discrete Math. 2:65-74.

Krarup, J., and Pruzan, P.M. (1978). Selected families of discrete location problems. "Discrete Optimization" (P. Hammer et al., eds.). Ann. Discrete Math. 4: (Also to be published as a book by North-Holland Pub. Co., Amsterdam.)

Kruskal, J.B. Jr. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Amer. Math. Soc. 7:48-50.

Kuhn, H.W. (1955). The Hungarian method for the assignment problem. Naval Res. Logistics Quart. 2:83-97.

Ladner, R. (1975). On the structure of polynomial time reducibility. J. ACM 22:155-171.

Las Vergnas, M. (1975). Matroides orientables. C.R. Acad. Sci. Paris 280:61-64.

Lawler, E.L. (1975). Matroid intersection algorithms. Math. Programming 9:31-56.

Lawler, E.L. (1976). "Combinatorial Optimization: Networks and Matroids." Holt, Rinehart and Winston, New York.

Lawler, E.L. (1978). Shortest path and network flow algorithms. In "Discrete Optimization" (P. Hammer et al., eds.). Ann. Discrete Math. 4:

Lawler, E.L., and Wood, D.E. (1966). Branch-and-bound methods: a survey. Operations Res. 14:699-719.

Lawrence, J. (1975). Oriented matroids. Ph.D. dissertation, Dept. of Math., Univ. of Washington, Seattle. To be published in J. Combinatorial Theory.

Lenstra, J., and Rinnooy Kan, A. (1975). Some simple applications of the traveling salesman problem. Operational Res. Qtly. 26:717-733.

Lin, S. (1965). Computer solutions of the traveling salesman problem. Bell System Tech. J. 44:2245-2269.

Lin, S., and Kernighan, B.W. (1973). An effective heuristic algorithm for the traveling salesman problem. Operations Res. 21:498-516.

Little, J.D.C., Murty, K., Sweeney, D.W., and Carel, C. (1963). An algorithm for the traveling salesman problem. Operations Res. 11:972-989.

Lovász, L. (1972). Normal hypergraphs and the perfect graph conjecture. Discrete Math. 2:253-267.

Lovász, L. (1977). Certain duality principles in integer programming. "Studies in Integer Programming" (P.L. Hammer et al., eds.). Ann. Discrete Math. 1:363-374.

Lueker, G.S. (1978). Maximization problems on graphs with edge weights chosen from a normal distribution. Proc. ACM Symp. Theory of Computing 10:13-18.

Minty, G.J. (1966). On the axiomatic foundations of the theories of directed linear graphs, electrical networks, and network programming. J. Math. Mech. 15:485-520.

Minty, G.J. (1978?). On maximal independent sets of vertices in claw-free graphs. J. Combinatorial Theory B.

Mitten, L.G. (1970). Branch-and-bound methods: general formulation and properties. Operations Res. 18:24-34.

Norman, R.Z., and Rabin, M.O. (1959). An algorithm for a minimum cover of a graph. Proc. Amer. Math. Soc. 10:315-319.

Papadimitriou, C.H. (1976). On the complexity of edge traversing. J. ACM 23:544-554.

Papadimitriou, C.H. (1977). The Euclidean traveling salesman problem is NP-complete. Theoretical Comput. Sci. 4:237-244

Papadimitriou, C. H., and Steiglitz, K. (1977). On the complexity of local search for the traveling salesman problem. SIAM J. Comput. 6:76-83.

Pape, U. (1977). Eine Bibliographie zu "Kürzeste Weglängen und Wege in Graphen." Bericht 77-07, Fachbereich 20, Technische Universität Berlin.

Pierce, A. R. (1976). Bibliography on algorithms for shortest path, shortest spanning tree and related circuit routing problems. (1956-1974). Networks 5:129-149.

Pósa, L.(1976). Hamiltonian circuits in random graphs. Discrete Math. 14:359-364.

Prim, R. C. (1957). Shortest connection networks and some generalizations. Bell System Tech. J. 36:1389-1401.

Rabin, M. O. (1972). Proving simultaneous positivity of linear forms. J. Comput. Systems Sci. 6:639-650.

Rado, R. (1957). Note on independence functions. Proc. London Math. Soc. 7:300-320.

Raymond, T. C. (1969). Heuristic algorithm for the traveling salesman problem. IBM J. Res. Dev. 13:400-407.

Reiter, S., and Sherman, G.S. (1965). Discrete optimizing. SIAM J. Appl. Math. 13:864-889.

Rivest, R.L., and Vuillemin, J. (1975). A generalization and proof of the AAnderaa/Rosenberg conjecture. Proc. ACM Symp. Theory of Computing 7:6-11.

Rockafellar, R.T. (1969). The elementary vectors of a subspace of R^N. In "Combinatorial Mathematics and its Applications," Proc. Chapel Hill Conf. (R.C. Bose and T.A. Dowling, eds.), pp. 104-127.

Sahni, S., and Gonzales, T. P-complete approximation problems. J. ACM 23:555-565.

Savage, S.L., Weiner, P., and Bagchi, A. (1976). Neighborhood search algorithms for guaranteeing optimal traveling salesman tours must be inefficient. J. Comput. System Sci. 12:25-35.

Shamos, M. I., and Hoey, D. (1975). Closest point problems. Proc. IEEE Symp. Foundations of Computer Science 17:268-273.

Shapiro, D.M. (1966). Algorithms for the solution of the optimal cost and bottleneck traveling salesman problems. Sc.D. Dissertation, Washington Univ., St. Louis.

Smith, T.H.C., Srinivasan, V., and Thompson, G.L. (1977). Computational performance of three subtour elimination algorithms for solving asymmetric traveling salesman problems. "Studies in Integer Programming" (P. L. Hammer et al., eds.). Ann. Discrete Math. 1:495-506.

Smith, T.H.C., and Thompson, G.L. (1977). A LIFO implicit enumeration search algorithm for the symmetric traveling salesman problem using Held and Karp's 1-tree relaxation. "Studies in Integer Programming" (P.L. Hammer et al., eds.). Ann. Discrete Math. 1:479-493.

Spira, P.M. (1972). Complete linear proofs of systems of linear inequalities. J. Comput. Systems Sci. 6:205-216.

Spira, P.M. (1973). A new algorithm for finding all shortest paths in a graph of positive arcs in average time $O(n^2 \log^2 n)$. SIAM J. Comput. 2:28-32.

Spira, P.M., and Pan, A. (1975). On finding and updating spanning trees and shortest paths. SIAM J. Comput. 4:375-380.

Strassen, V. (1969). Gaussian elimination is not optimal. Numer. Math. 13:354-356.

Sysło, M.M. (1973). A new solvable case of the traveling salesman problem. Math. Programming 4:347-348.

Tanimoto, S.L. (1976). Analysis of biomedical images using maximal matching. Proc. 1976 IEEE Conf. on Decision and Control, pp. 171-176.

Tarjan, R.E. (1977). Finding optimum branchings. Networks 7:25-35.

Tomizawa, N. (1972). On some techniques useful for solution of transportation problems. Networks 1:173-194.

Wagner, R.A. (1976). A shortest path algorithm for edge-sparse graphs. J. ACM 23:50-57.

Walkup. D.W. (1977a). On the expected value of a random assignment problem. Tech. Report, Dept. of Computer Sci., Washington Univ., St. Louis.

Walkup, D.W. (1977b). Matchings in random regular bipartite graphs. Tech. Report, Dept. of Comput. Sci., Washington Univ., St. Louis.

Weide, B. (1977). A survey of analysis techniques for discrete algorithms. Computing Surveys 9:291-313.

Welsh, D.J. (1968). Kruskal's theorem for matroids. Proc. Cambridge Phil. Soc. 64:3-4.

Welsh, D.J.A. (1976). Matroid Theory. Academic Press, London-New York.

Whitehouse, G.E. (1973). "Systems Analysis and Design Using Network Techniques." Prentice-Hall, Englewood Cliffs, New Jersey.

Whitney, H. (1935). On the abstract properties of linear dependence. Amer. J. Math. 57:509-533.

Yao, A.C. (1975). An $O(|E| \log \log |V|)$ algorithm for finding minimum spanning trees. Information Processing Letters 4:21-23.

Yao, A.C. (1976). On the complexity of comparison problems using linear functions. Proc. IEEE Symp. Foundations of Computer Science 16:85-89.

Yao, A.C., Avis, C.M., and Rivest, R.L. (1977). An $\Omega(n^2 \log n)$ lower bound to the shortest paths problem. Proc. ACM Symp. Theory of Computing 9:11-17.

Yen, J.Y. (1972). Finding the lengths of all shortest paths in N-node nonnegative-distance complete networks using $\frac{1}{2}N^3$ additions and N^3 comparisons. J. ACM 19:423-424.

Yuval, G. (1976). An algorithm for finding all shortest paths using $N^{2.81}$ *infinite-precision* multiplications. Information Processing Letters 4:155-156.

Zadeh, N. (1973). A bad network problem for the simplex method and other minimum cost flow algorithms. Math. Programming 5:255-266.